1. ***Busy chef.*** Construct NFAs that recognize the following languages.

   (a) Let *Sandwich* be an automatic language.

   $$Cut(Sandwich) = \left\{ sandwich \ : \ sandwich \cdot sandwich^R \in Sandwich \right\},$$

   where $sandwich^R$ denote the reversal of the string *sandwich*.

   Solution: Let M be a DFA that recognize the automatic language *Sandwich*, described by the tuple $(Q, s, A, \Sigma, \delta)$. We now construct a new NFA N, described by $(Q', S', A', \Sigma', \delta')$, that recognizes the language $Cut(Sandwich)$.

   - States Q': $Q \times Q \cup \{s'\}$ — pairs of states in Q, plus an extra starting state s'
   - Starting states S': $\{s'\}$
   - Accepting states A': $\{(r, r) \in Q' : \text{for any } r \in Q\}$
   - Alphabet Σ': $\Sigma \cup \{\varepsilon\}$ — allowing ε-transitions
   - Transition function δ':
     - We add a transition from (p, q) to (p', q') on reading symbol $a \in \Sigma$ if

       $$\delta(p, a) = p' \quad \text{and} \quad \delta(q', a) = q;$$

       in other words, we walk the DFA transition in the first copy of M *forwards* but the one in the second copy of M *backwards*. In notation,

       $$\delta'((p, q), a) = \left\{ (p', q') \in Q' : \delta(p, a) = p' \text{ and } \delta(q', a) = q \right\}$$

       for any $p, q \in Q$ and $a \in \Sigma$.
     - We also add ε-transitions from s to each state in $Q' = Q \times Q$ such that the first component is the starting state s of M and the second component is an accepting state in M. In notation,

       $$\delta'(s', \varepsilon) = \left\{ (s, q) \in Q' : q \in A \right\}.$$

   The intuition behind the construction of M is that we attempt to find an accepting path in M starting with two fingers, one pointing to the starting state of s and the other is at any of the accepting states in A. As we do not know where does the accepting path end, we use the ε-transitions to guess all the possible starts in the new NFA N. Now as we read the word *sandwich*, both fingers will move to the neighboring states in the path, one traversing forward in M and the other backward. Assuming the existence of an accepting path in M, the two fingers will eventually meet at the middle on a single state r, which the corresponding state (r, r) in the constructed NFA N is accepting. If there are no such accepting path in M, there is no way for the two fingers to meet in M and therefore we won't have a combination of a correct guess for the initial ε-transition and a word *sandwich* that leads to an accepting state in the constructed NFA N. ■

Rubric: It is good to remember that notations are not the only way to describe your constructions formally. With practice we can convey the same meaning using English. The important thing is that your readers should be able to recover your construction unambiguously from your writing *alone* (not with the help of figures/intuitions).

Standard 5-point grading scale plus deadly-sins and sudden-death rules. Constructions without English explanations receive an automatic ***zero***.

2. ***Prefix codes.*** ***Huffman code*** is an efficient lossless encoding method that achieves optimal (symbol-to-symbol) compression rate when the input probability distribution is known. The *prefix-free* property, that no codeword inside the constructed encoding is the prefix of another codeword, makes Huffman code uniquely decodable and very efficient in practice.

   Let L be an automatic language. Construct NFAs that recognize the following languages.

   (a) $\{w \in \Sigma^* : \text{no proper prefix of } w \text{ is in } L\}$

   Solution: Let M be a DFA that recognize the automatic language L, described by the tuple $(Q, s, A, \Sigma, \delta)$. We now construct a new NFA N, described by $(Q', S', A', \Sigma', \delta')$, that recognizes the language in Problem 2(a).

   - States Q': Q
   - Starting states S': $\{s\}$
   - Accepting states A': Q'
   - Alphabet Σ': Σ
   - Transition function δ': We remove a transition from p to q if p is an accepting state in Q. In notation,
     $$\delta'(p, a) = \varnothing$$
     for any $p \in A$ and $a \in \Sigma$.

   The intuition behind the construction is that no accepting path can pass through another accepting state in the midway, because we drop all the outgoing transitions from the all the accepting states. ■

   (b) $\{w \in \Sigma^* : \text{no proper suffix of } w \text{ is in } L\}$

   Solution: First we notice that the language stated in Problem 2(b), denoted as L^{2b}, is equivalent to
   $$\{w \in \Sigma^* : \text{no proper prefix of } w^R \text{ is in } L^R\},$$
   which then is equivalent to
   $$\{w^R \in \Sigma^* : \text{no proper prefix of } w \text{ is in } L^R\}.$$
   In other words, L^{2b} is equivalent to the reverse of the language from Problem 2(a) but applying on L^R. Denote the language from Problem 2(a) applying on L^R as $(L^R)^{2a}$. We have
   $$L^{2b} = ((L^R)^{2a})^R.$$

   From the working session we saw that the reverse of an automatic/regular language is still automatic/regular; in particular, L^R is regular. From the NFA constructed in Problem 2(a) applying on L^R, we know that the language $(L^R)^{2a}$ is regular, and therefore L^{2b}, the reverse of $(L^R)^{2a}$, is also regular. ■

Rubric: You can imagine each of the closure properties as a powerful kitchen tool. The more tools you learned, the more food materials you can process with ease and therefore the more complicated the dishes you get to cook. There are times when one has to invent a new tool in order to continue; in the context of regular languages, it often boils down to building some clever NFAs or using the subset construction. Rarely you will encounter a dish like Consommé or Turducken that requires you to give your all (like, in Problem 1(c)).

Standard 5-point grading scale plus deadly-sins and sudden-death rules. Constructions without English explanations receive an automatic ***zero***.